

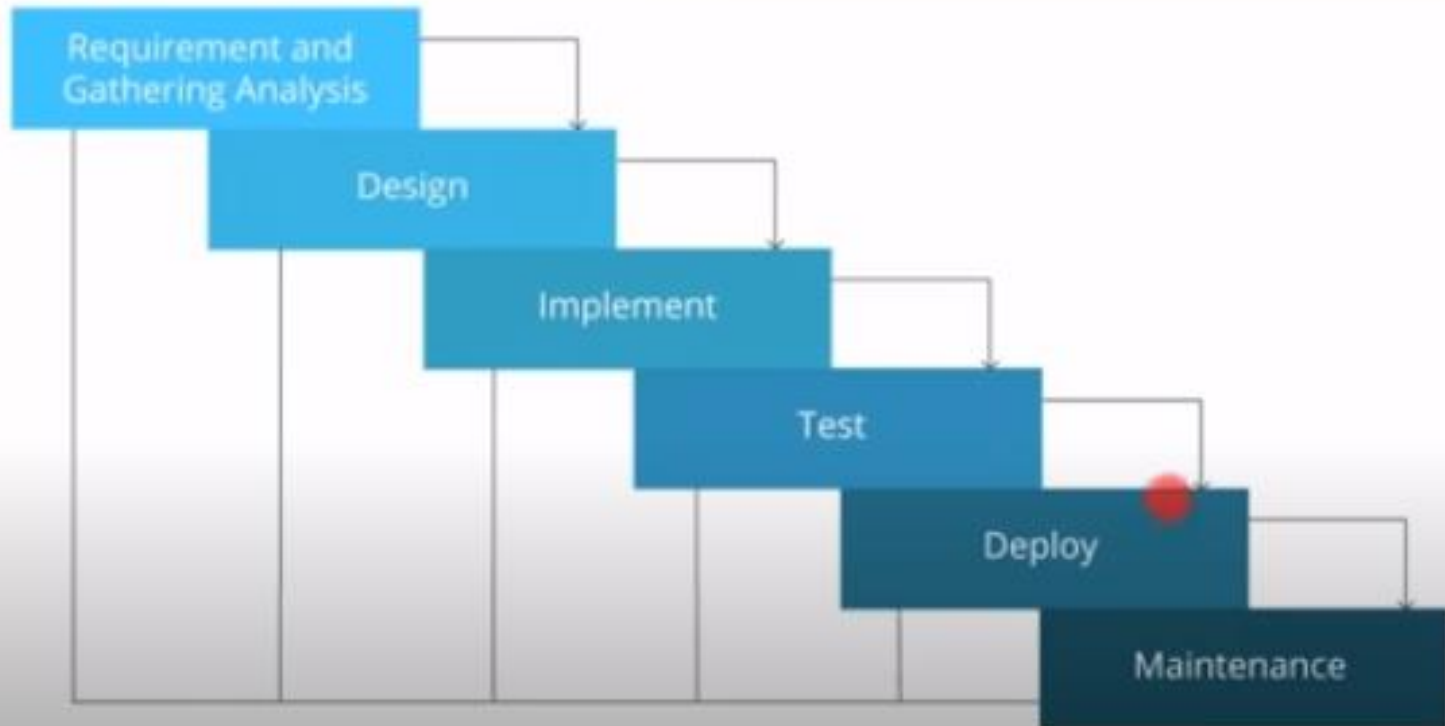
Coverage

- Devops overview
- Version control using git
- Build Maven
- Continuous integration using Jenkins.
- Automation Testing using Selenium
- Containerization using docker
- Monitoring

- Git
- Maven
- Jenkins
- Docker
- Selenium
- BDD

- Agenda
 - what is devops
 - what is devops?
 - Devops stages?
 - Tools
 - old methodology
-
- Let us start with watrefall model:
 - before devops organization was using this methodology

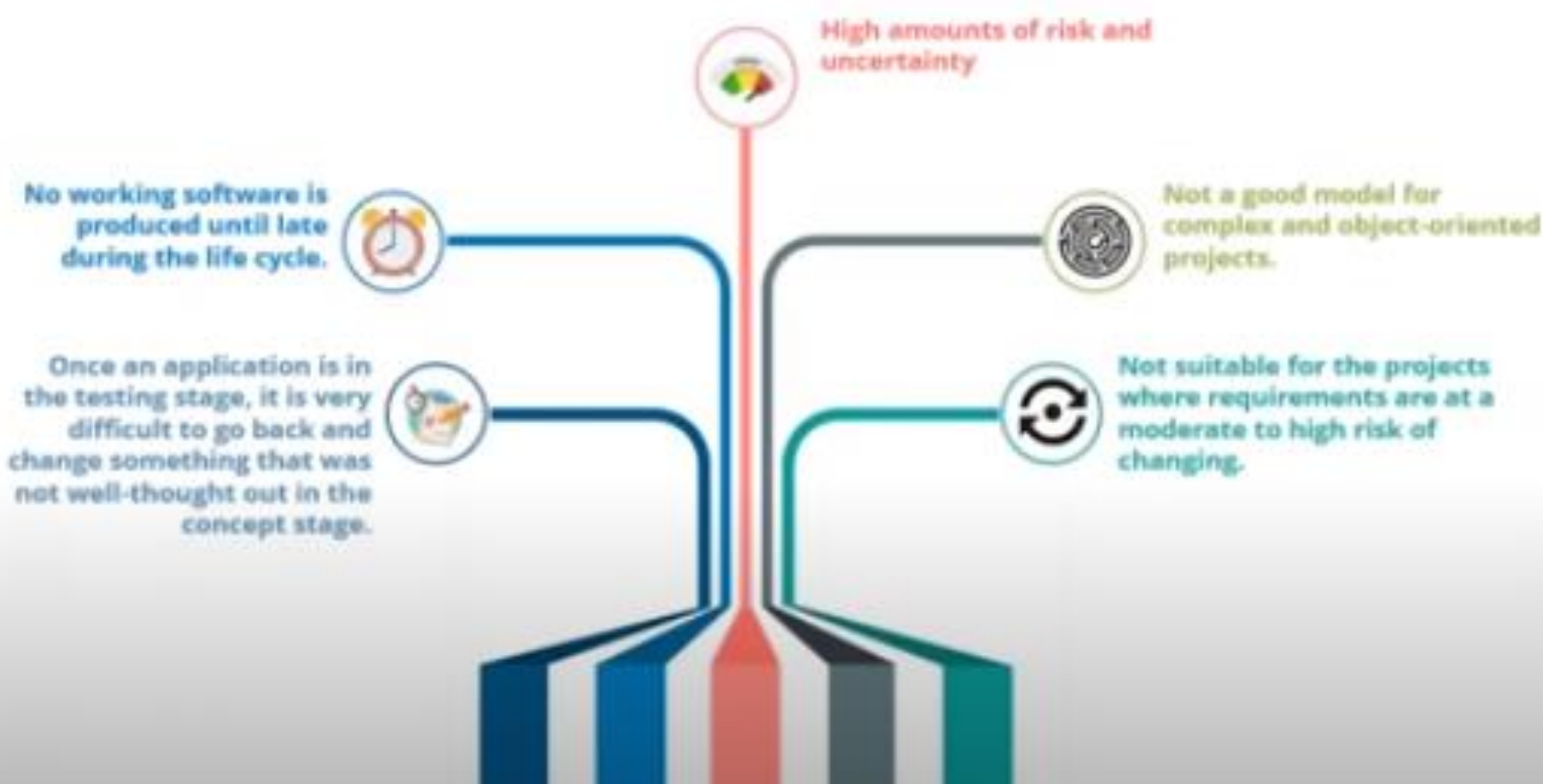
Traditional Waterfall Model



- Build include compliation,unit testing ,packaging .
- Deployed on test server for testing and then on production server for release.
- After release the application is monitor.

- Explain in brief the waterfall model.
- Various stages
- Linear and sequential approach

ations of Waterfall Model



- Difficult to do changes
- Difficult to debug the the error at late stages.
- Time consuming
- no deliverable , it is in late stage of life cycle.
- Not good for complex and object oriented project.
- Chances of change in technology

- Agile Methodology

What is Agile Methodology?

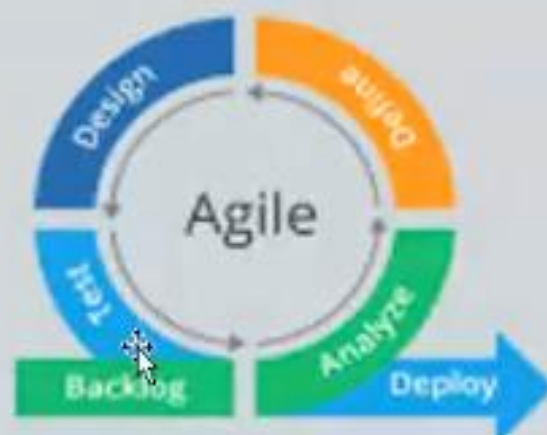
In the Agile Methodology, each project is broken up into several 'Iterations'

All Iterations should be of the same time duration (between 2 to 8 weeks)

At the end of each iteration, a working product should be delivered



Waterfall vs. Agile



- Analyze→plan→design→build→test→deploy
- Each iteration will return deliverable product
- Once the final testing is done it will be deployed in prod server.
- Development is continuous but deployment is not .
- Deployment is linear and manual where as dev is continuous.

- Agile
- Continuous dev and testing in each iteration
- Multiple iteration
- Design test and build is continuous
- Limitation:
- Dev part is agile and dynamic but the deployment is not .
- Agile is linear but deployment is still linear and manual
- Deployment need stability.

- Continuous iteration of dev and testing throughout the life cycle of soft development.
- Dev+testing→release of iteration (working product)

- Solution is devops
- In agile dev and testing is continuous but when we talk abt the deployment it is not continuous. For example the code working on developer laptop but not in production enviroment.
- Developer wants the agility where as operation team wants stability. To overcome this devops came into picture .
- ops team take care of app deployment,server maintenance,monitoring etc.

What is DevOps?



+



Developers & Testers

IT Operations

- What is DevOps?
- The DevOps is a combination of two words, one is software Development, and second is Operations.
- This allows a single team to handle the entire application lifecycle, from development to **testing, deployment, and operations**.
- DevOps helps you to reduce the disconnection between software developers, quality assurance (QA) engineers, and system administrators.

- Why DevOps?
- Before going further, we need to understand why we need the DevOps over the other methods.
- The operation and development team worked in complete isolation.
- After the design-build, the testing and deployment are performed respectively. That's why they consumed more time than actual build cycles.
- Without the use of DevOps, the team members are spending a large amount of time on designing, testing, and deploying instead of building the project.
- Manual code deployment leads to human errors in production.
- Coding and operation teams have their separate timelines and are not in synch, causing further delays.

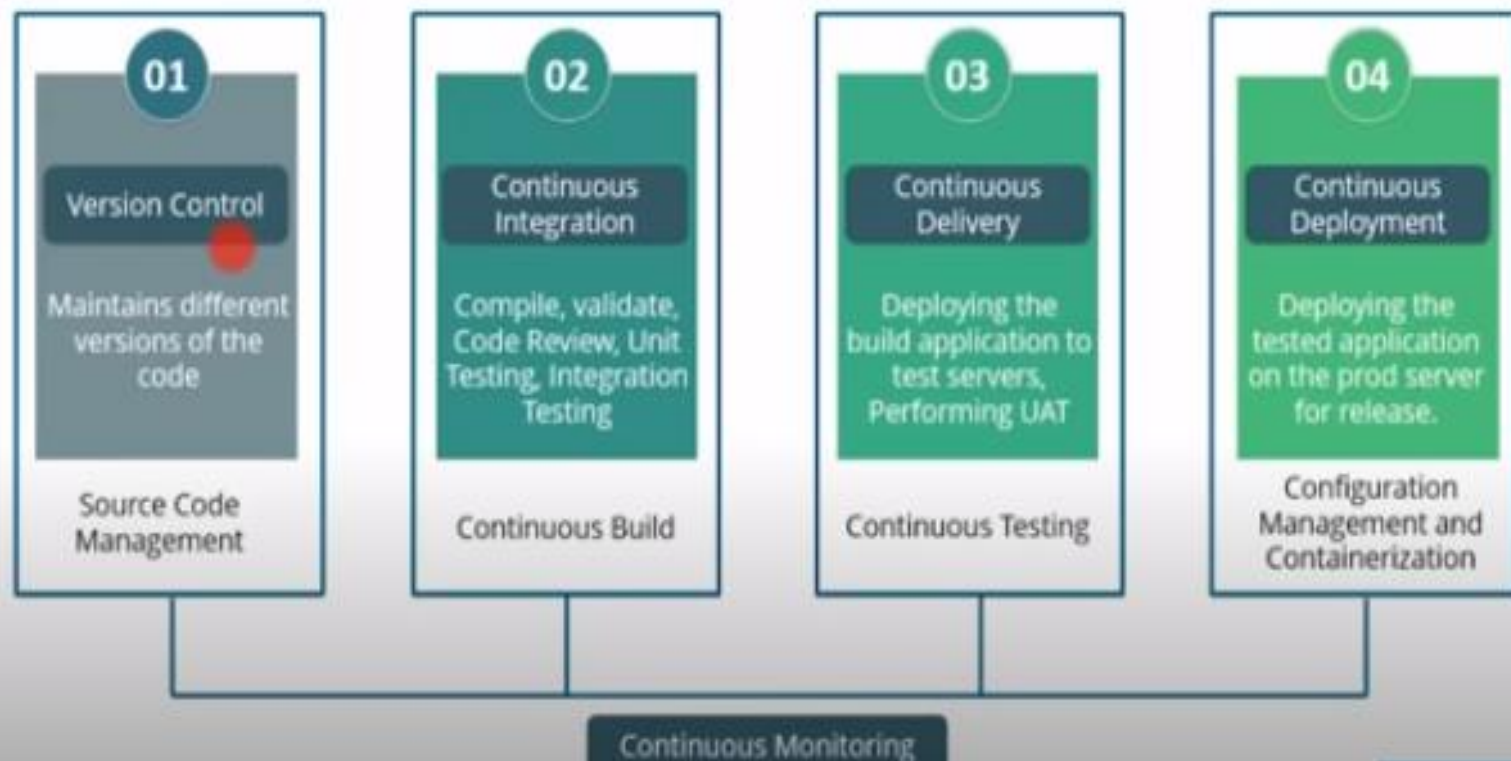
- DevOps History
- In 2009, the first conference named **DevOpsdays** was held in Ghent Belgium. Belgian consultant and Patrick Debois founded the conference.
- In 2012, the state of DevOps report was launched and conceived by Alanna Brown at Puppet.
- In 2014, the annual State of DevOps report was published by Nicole Forsgren, Jez Humble, Gene Kim, and others. They found DevOps adoption was accelerating in 2014 also.
- In 2015, Nicole Forsgren, Gene Kim, and Jez Humble founded DORA (DevOps Research and Assignment).
- In 2017, Nicole Forsgren, Gene Kim, and Jez Humble published "Accelerate: Building and Scaling High Performing Technology Organizations".

- DevOps Architecture Features
- Here are some key features of DevOps architecture, such as:



- Automation
 - Automation can reduce time consumption, especially during the testing and deployment phase. The productivity increases, and releases are made quicker by automation
- Collaboration
 - The Development and Operations team collaborates as a DevOps team, which improves the cultural model as the teams become more productive with their productivity, which strengthens accountability and ownership.
- Integration
 - Applications need to be integrated with other components in the environment. The integration phase combines the new functionality and then tested. Continuous integration and testing enable continuous development.
- Configuration management
 - It ensures the application to interact with only those resources that are concerned with the environment in which it runs. The configuration files are not created where the external configuration to the application is separated from the source code. The configuration file can be written during deployment, or they can be loaded at the run time, depending on the environment in which it is running.

DevOps Stages



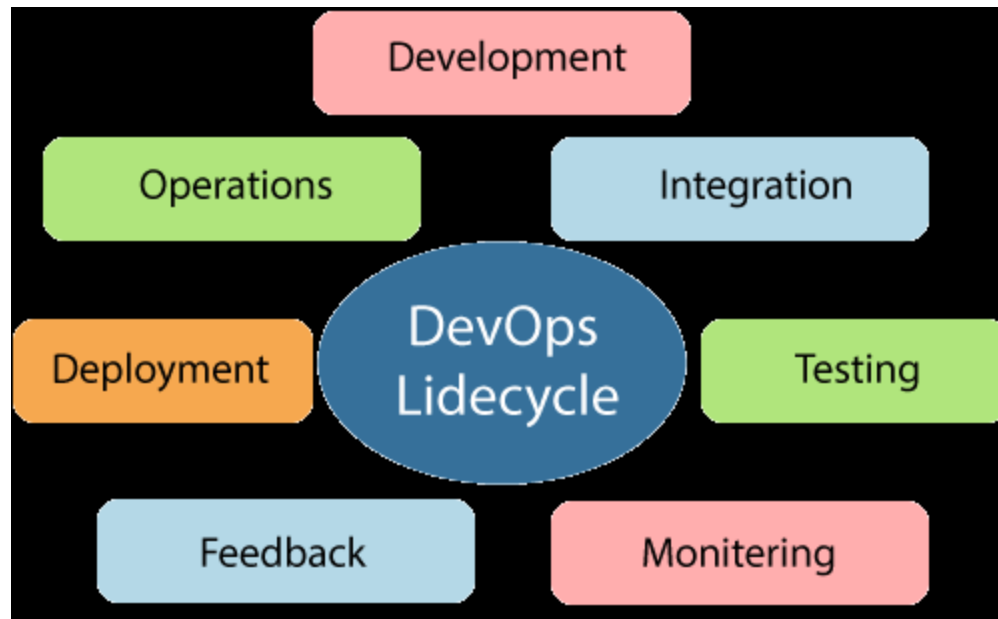
- Version control-Git GitHub(SCM)
- Integration—Jenkins, Building code is not only involved compilation, but it also includes **unit testing, integration testing, code review, and packaging.**
- Continuous Delivery (continuous testing)—once the build is done jenkins will deploy the app on test server, for unit testing ,UAT, functional once done
- Continuous Deployment (Configuration mgt and containerization)—tested build will be deployed on prod server,maven ,gradle,

The operations team job is to test the code, and provide feedback to developers in case of bugs. If all goes well, the operations team uploads the code to the build servers



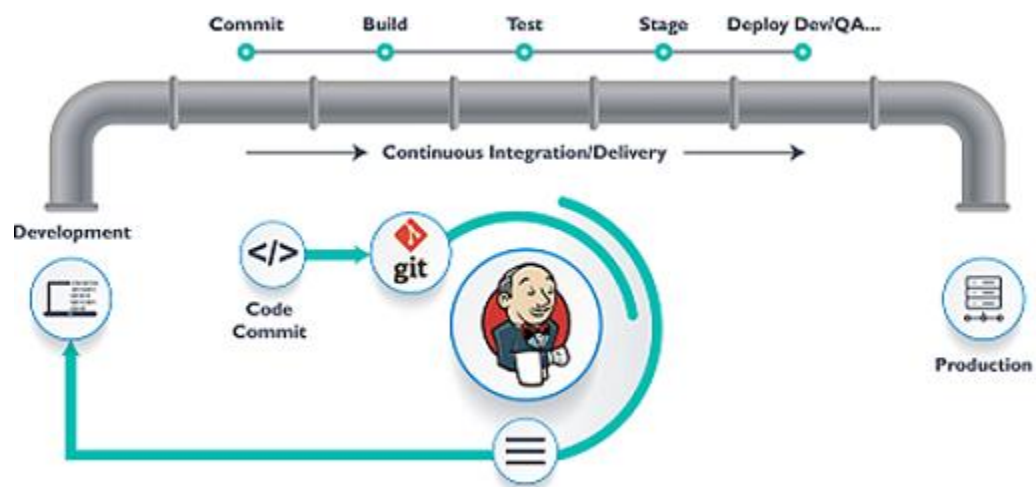
Operations

- Docker/Kubernetes –Containerization orchestration platform
- Puppet and ansible-configuration mgt
- s/w testing –selenium
- Integration –jenkins
- Nagios-continuous monitoring



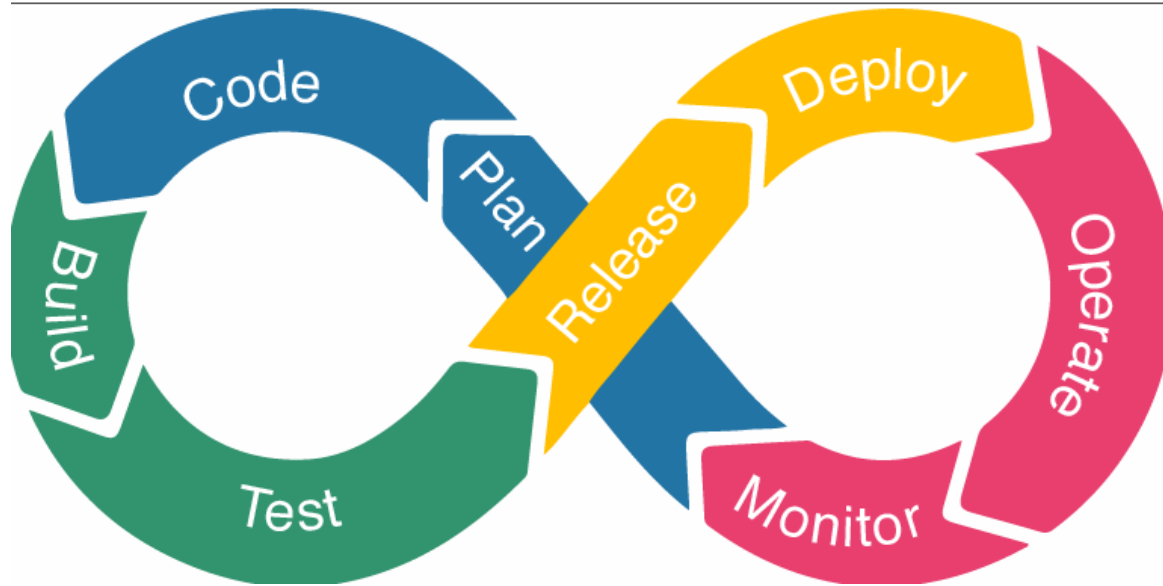
- DevOps Lifecycle
- DevOps defines an agile relationship between operations and Development. It is a process that is practiced by the development team and operational engineers together from beginning to the final stage of the product.
- Learning DevOps is not complete without understanding the DevOps lifecycle phases. The DevOps lifecycle includes seven phases as given below:

- Continuous development: This phase involves the planning and coding of the software. The vision of the project is decided during the planning phase. And the developers begin developing the code for the application. There are no DevOps tools that are required for planning, but there are several tools for maintaining the code.
- Continuous Integration
- This stage is the heart of the entire DevOps lifecycle. It is a software development practice in which the developers require to commit changes to the source code more frequently. This may be on a daily or weekly basis. Then every commit is built, and this allows early detection of problems if they are present. Building code is not only involved compilation, but it also includes **unit testing, integration testing, code review, and packaging**. The code supporting new functionality is continuously integrated with the existing code. Therefore, there is continuous development of software. Jenkins is a popular tool used in this phase. Whenever there is a change in the Git repository, then Jenkins fetches the updated code and prepares a build of that code, which is an executable file in the form of war or jar. Then this build is forwarded to the test server or the production server.



- 3) Continuous Testing
- This phase, where the developed software is continuously testing for bugs. For constant testing, automation testing tools such as **TestNG**, **JUnit**, **Selenium**, etc are used
- 4) Continuous Monitoring
- Monitoring is a phase that involves all the operational factors of the entire DevOps process, where important information about the use of the software is recorded and carefully processed to find out trends and identify problem areas. Usually, the monitoring is integrated within the operational capabilities of the software application.
- 5) Continuous Feedback
- The application development is consistently improved by analyzing the results from the operations of the software. This is carried out by placing the critical phase of constant feedback between the operations and the development of the next version of the current software application.

- 6)Continuous Deployment
- In this phase, the code is deployed to the production servers. Also, it is essential to ensure that the code is correctly used on all the servers.
- Let us try to understand a few things about Configuration management and [Containerization tools](#). These set of tools here help in achieving Continuous Deployment (CD).
- [Configuration Management](#) is the act of establishing and maintaining consistency in an application's functional requirements and performance. There are several components in a configuration management system. Managed systems can include servers, storage, networking, and software. Let me put this in simpler words, it is the act of releasing deployments to servers, scheduling updates on all servers and most importantly keeping the configurations consistent across all the servers.
- Since the new code is deployed on a continuous basis, configuration management tools play an important role in executing tasks quickly and frequently. Some popular tools that are used here are Puppet, [Chef](#), [SaltStack](#), and [Ansible](#).
- Containerization tools also play an equally important role in the deployment stage. Docker and Vagrant are the popular tools used for this purpose. These tools help produce consistency across Development, Test, Staging and Production environments. Besides this, they also help in scaling-up and scaling-down of instances swiftly.
- Containerization tools help in maintaining consistency across the environments where the application is developed, tested and deployed. Using these tools, there is no scope of errors/ failure in the production environment as they package and replicate the same dependencies and packages used in the development/ testing/ staging environment. It makes your application easy to run on different computers.
- Containerization is the process of packaging an application along with its required libraries, frameworks, and configuration files together so that it can be run in various computing environments efficiently. In simpler terms, containerization is the encapsulation of an application and its required environment.
- Docker is a container management service. The keywords of Docker are develop, ship and run anywhere. The whole idea of Docker is for developers to easily develop applications, ship them into containers which can then be deployed anywhere.
- 7) Continuous Operations
- All DevOps operations are based on the continuity with complete automation of the release process and allow the organization to accelerate the overall time to market continually.



DevOps Phases and Tools



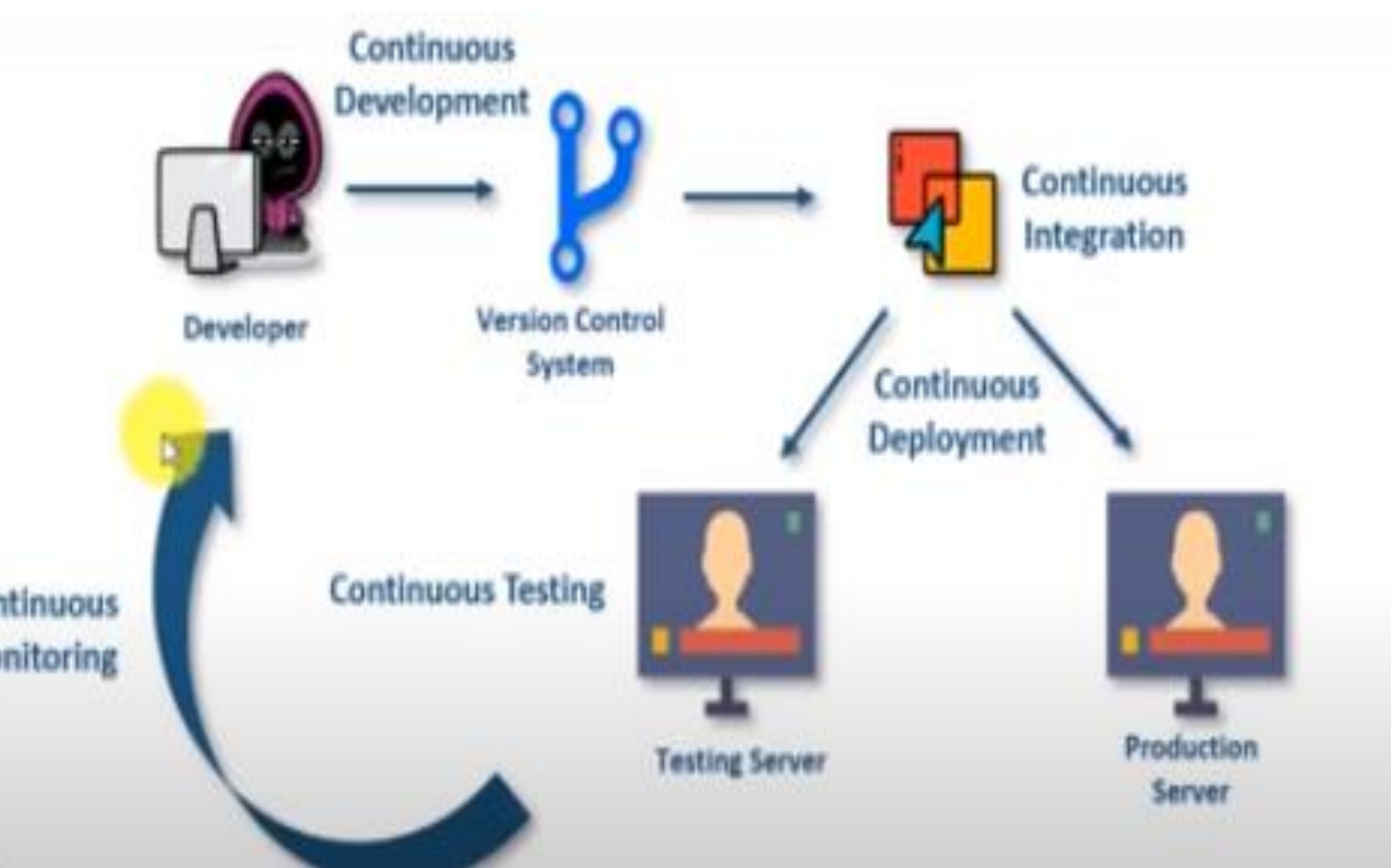
- Ex, developer develops some code for web app . We need to deploy it on application server to make it available to the end user.
- UI developer ,backend developer,db developer .will keep their code at central location called as source code mgt tools.
- Git—version controlling
- Github-source code management tool
- Convert code into artifact jar/war/rar format for this we need build tool such ant,gradle,maven etc
- Artifact will be loaded on server ex tomcat.
- Automate the things we use jenkins

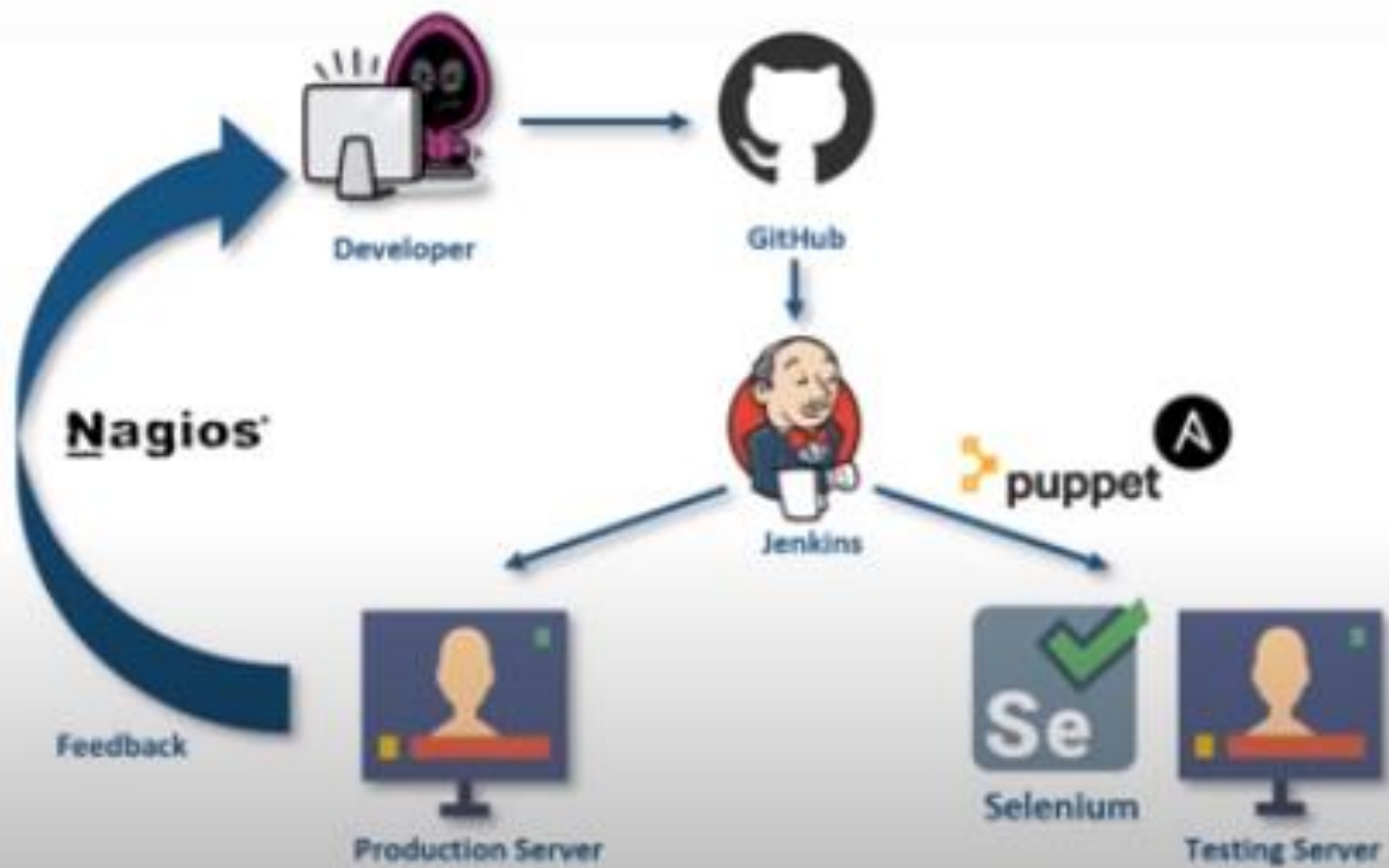
- The tested build is deployed on production environment
 - Operation-using the software, to see the traffic, execution,
 - Monitoring the functionality ,if not send the feedback and again start from plainning. The popular tools used for this are [Splunk](#), [ELK Stack](#), [Nagios](#), NewRelic and Sensu.
 - Life cycle is automated through tool.
-
- Docker is a container management service. The keywords of Docker are **develop**, **ship** and **run** anywhere. The whole idea of Docker is for developers to easily develop applications, ship them into containers which can then be deployed anywhere.
 - Docker will give the tool to the op team.developer will put the code inside docker and oush it to the next phase.

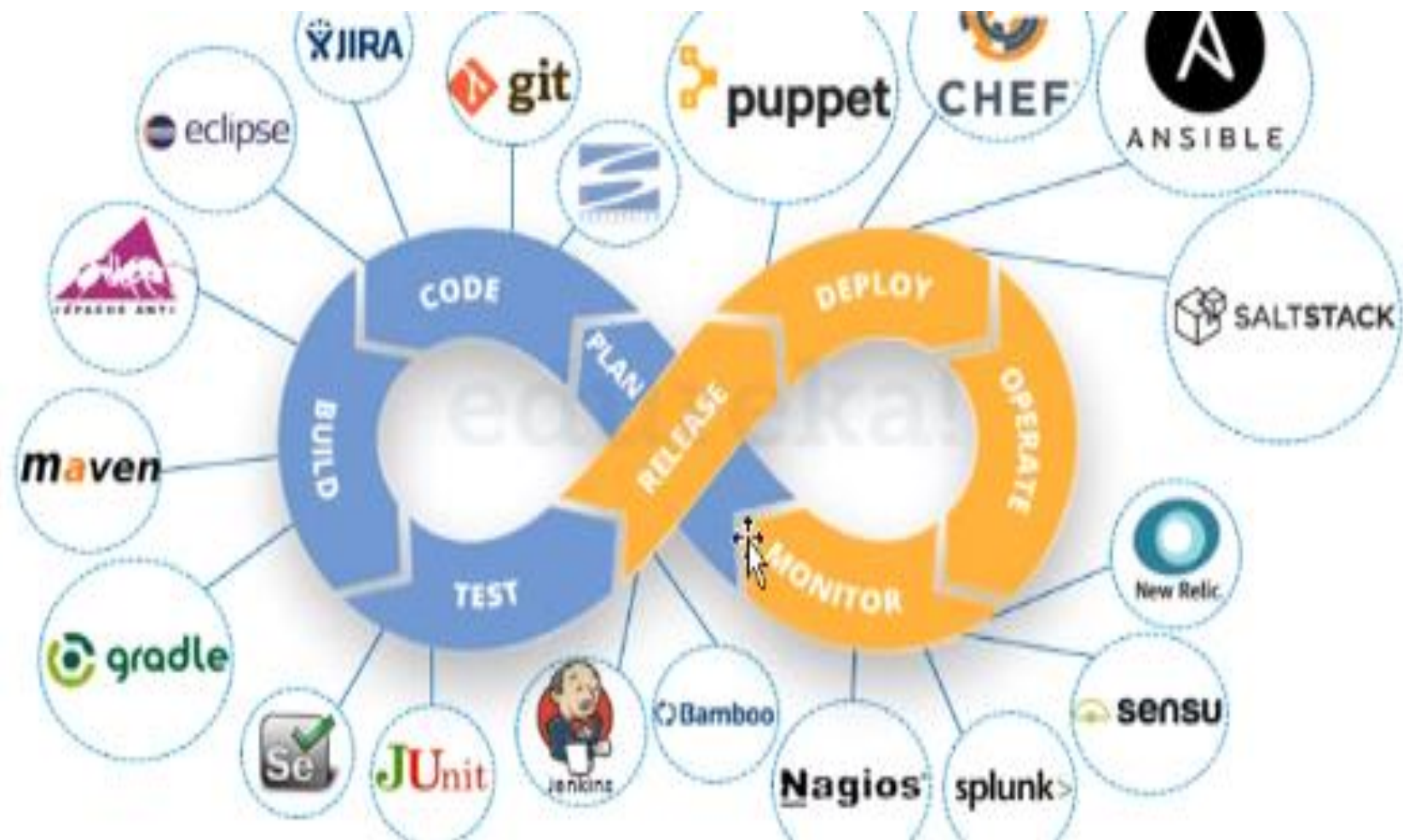
- PLANNING: requirement analysis
- Code: code version management , source management. Tools used git, git hub
- Build: compilation, validation, pkging unit testing, integration testing tool used maven ,gradle,ant etc.
- Test: build application is deployed in test environment for testing such as unit , end user,function testing , selenium JUnit,
- Deploy and operate: deployed on production environment.docker,puppet,chef etc.
- Monitor-continuous monitoring is done nagios,splunk
- Integration:heart of devops,jenkins. integration:building the application continuously if it has changed.compliling,packaging . Jenkins.any changes in the code in shared repos , jenkins will pull the changes and build the changes . build will be deployed on test server.deploy it on prod server and then finally by tools such as nagios,Splunk

The Devops Lifecycle divides the SDLC lifecycle into the following stages:









- CON DEPLOYMENT –
- Visualization & containerization:prb of sw running on differ platforms is solved by containerization, container will wrap the code with all.if we run the container on any sys it will run.container is the executable form of os
- Conf mgt:puppet ansible,no need to install the sw if there is sw comatibility issue ,
- Docker provides container

Stages in devops

- Version controlling—multiple developer is working on the same application to know about the which dev has made which commit and at what time if any issue then how we will revert to the dev .managing the source code. Like git
- Continuous integration:building the application continuously if it has changed.compiling,packaging . Jenkins
- Continuous delivery Jenkins .build is deployed on test server for testing User acceptance testing for that we use tools like selenium for automation testing
- Continuous deployment: Configuration mgt.

- New code → continuous integration server will pull and make a build 1 → 2
- 2 → 3 build will be deployed on test server
- 3 → 4 continuous deployment
- Monitor by continuous monitoring tool such as nagios
- Continuous deployment is not advisable bcoz before releasing we want to do multiple checks, testing, we don't want it shd be automated so we go for manual deployment tool such as puppet, docker, chef etc.

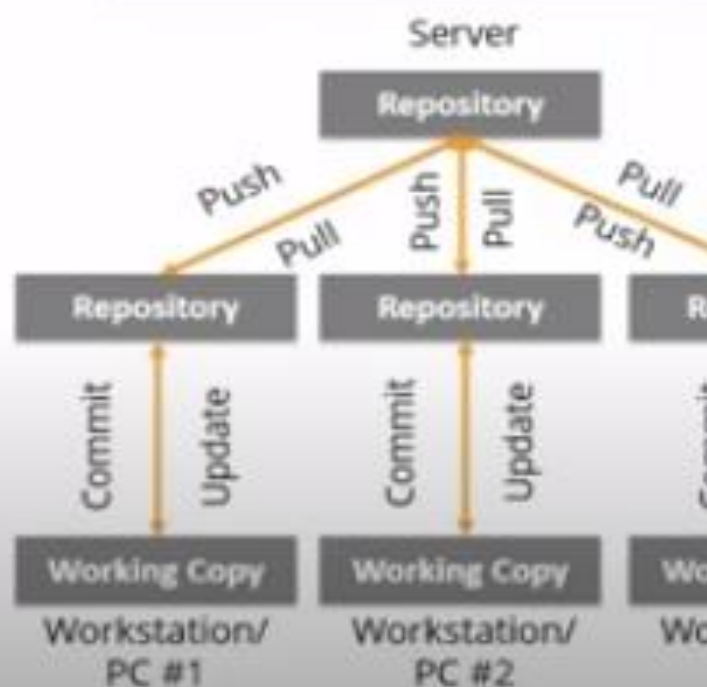
Source Code Management

The management of changes to documents, computer programs, large websites and other collection of information

Centralized Version Control System



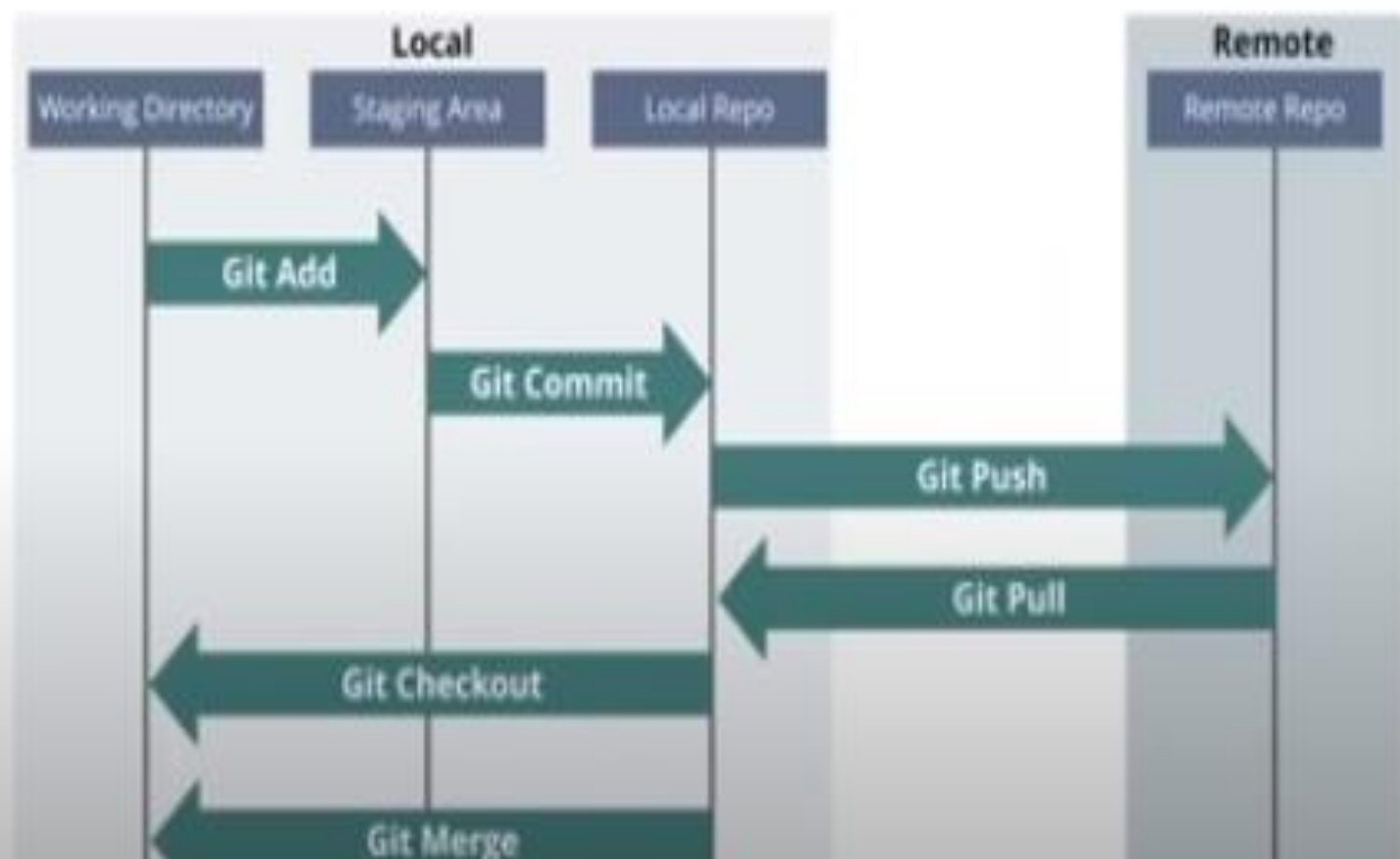
Distributed Version Control System



Source Code Management

git

Git is a Distributed Version Control tool that supports distributed non-linear workflows by providing data assurance for developing quality software



- Git command
- Git version
- =====Directory Command=====
- mkdir <folder name>
- cd <path of the directory>
- =====
- Dell@Admin MINGW64 /g/ManishaGit
- \$ mkdir Myprojct
- Dell@Admin MINGW64 /g/ManishaGit
- \$ cd Myprojct
- Dell@Admin MINGW64 /g/ManishaGit/Myprojct
- \$
- =====
- To convert directory into local repository
- Dell@Admin MINGW64 /g/ManishaGit/Myprojct
- \$ git init
- =====

- Add and Commit
- -----
- create a file in Myprojct
- example File1.txt
- add some text in it
- Now save the changes in staging error
- Dell@Admin MINGW64 /g/ManishaGit/Myprojct (master)
- \$ git add File1.txt
- will add the File1.txt to staging Area.

- To add all files in the staging error use
- -----
- Dell@Admin MINGW64 /g/ManishaGit/Myprojct (master)
- \$ git add .
- .-->Period
- \$git add File*
- \$ git status
- To commit the files in Local Repository
- -----
- Dell@Admin MINGW64 /g/ManishaGit/Myprojct (master)
- \$ git commit -m "First Commit"
- [master (root-commit) 1aef2b1] First Commit
- 1 file changed, 1 insertion(+)
- create mode 100644 File1.txt

DevOps Tutorial for Beginners | L x

Manisha-atos/gittest x

Rediffmail x

+

github.com/Manisha-atos/gittest/tree/master

Manisha-atos / gittest

Unwatch 1

Star 0

Fork 0

Code

Issues 0

Pull requests 0

Actions

Projects 0

Wiki

Security 0

Insights

Settings

No description, website, or topics provided.

Edit

Manage topics

1 commit

1 branch

0 packages

0 releases

1 contributor

Branch: master

New pull request

Create new file

Upload files

Find file

Clone or download

Manisha-atos Create readme.txt

readme.txt Create readme.txt

readme.txt

hello
welcome to git

Clone with HTTPS

Use SSH

Use Git or checkout with SVN using the web URL.

https://github.com/Manisha-atos/gittest.

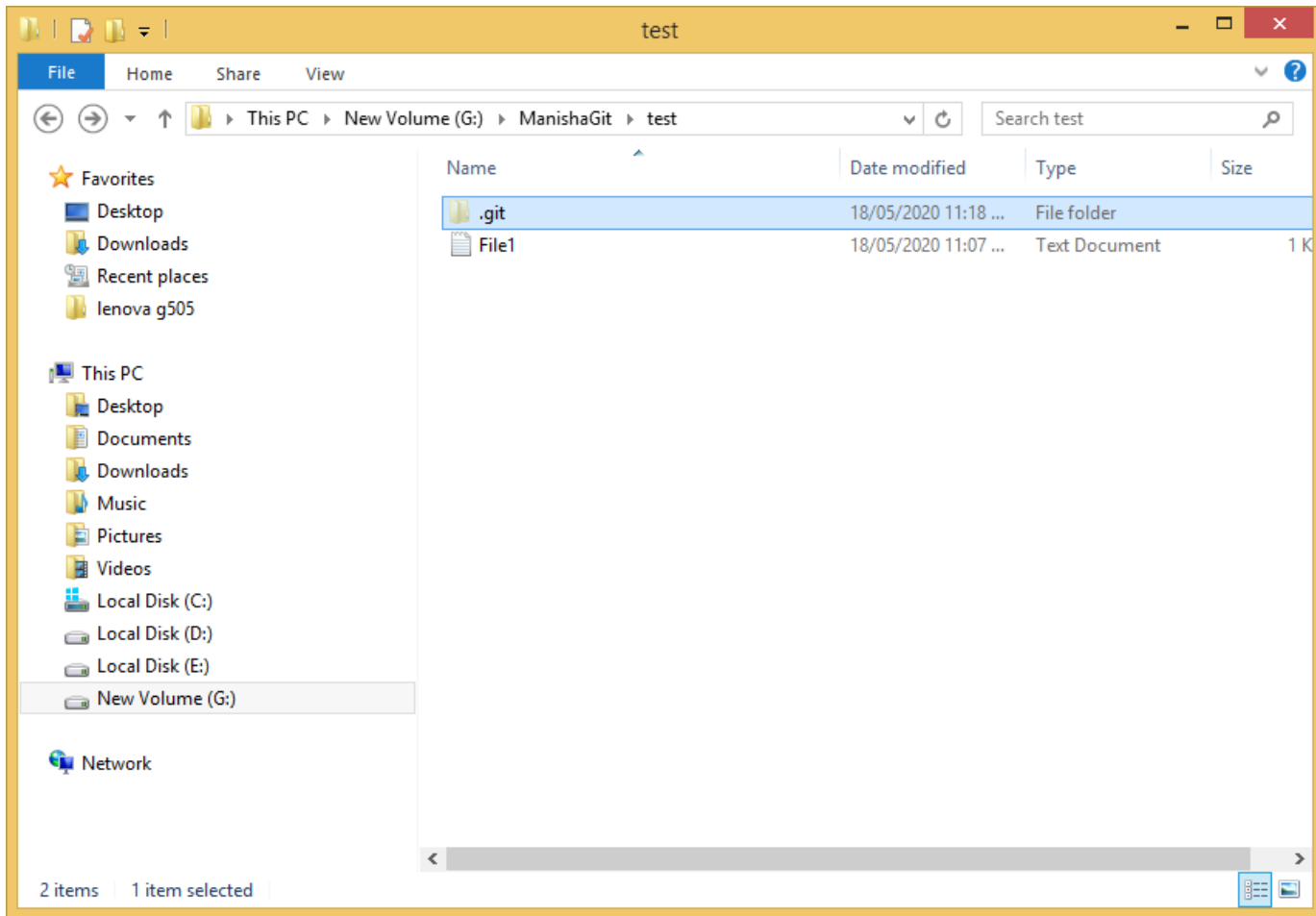
Open in Desktop

Download ZIP


Windows Taskbar

11:16 PM 18/05/2020

- Dell@Admin MINGW64 /g/ManishaGit/test (master)
- \$ git remote add origin <https://github.com/Manisha-atos/gittest.git>
- Will add the remote rep to locl repo



- To pull from remote rep to local rep
- \$git pull origin master
- git pull origin master --allow-unrelated-histories

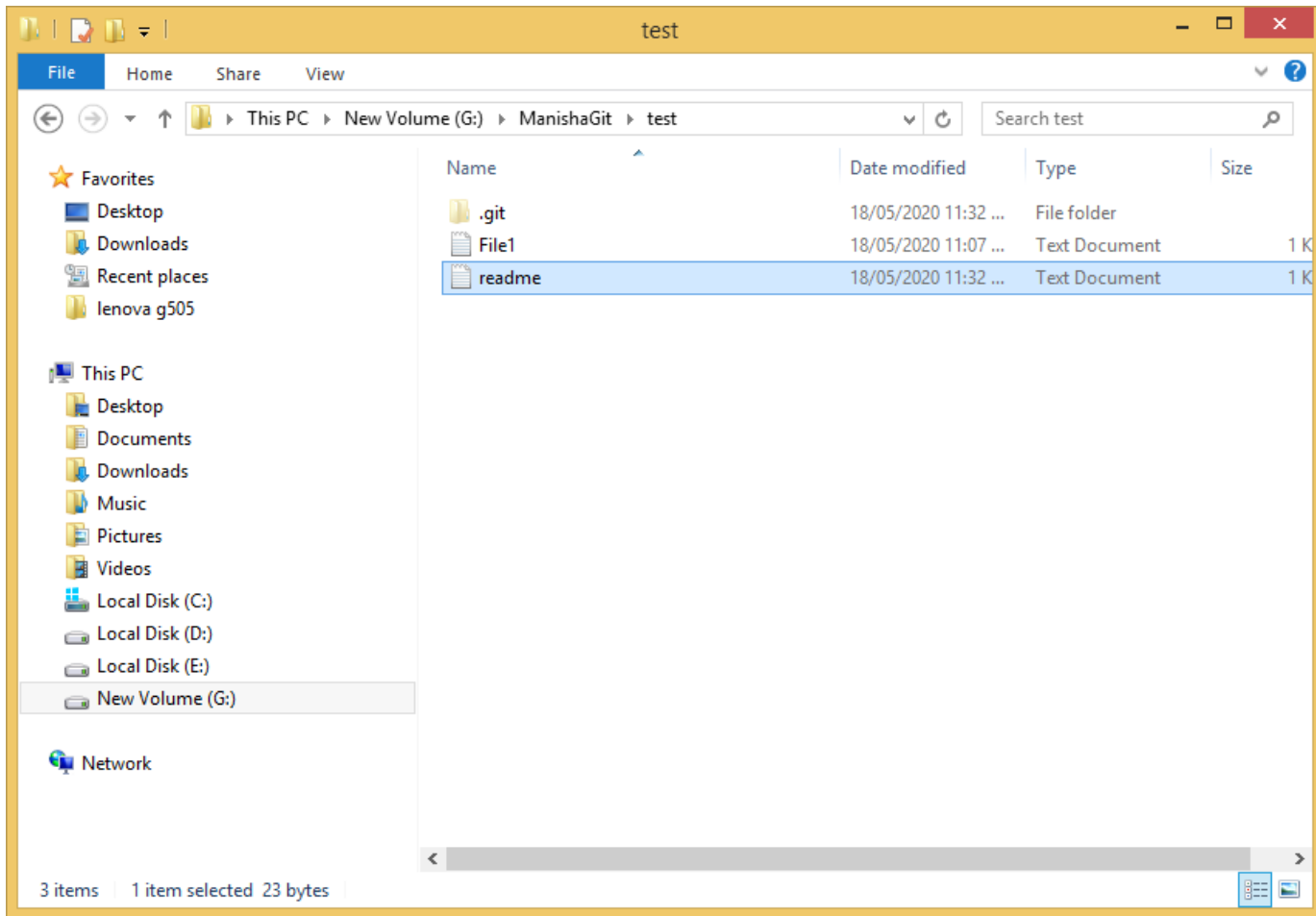


```
MINGW64:/g/ManishaGit/test
--mirror[=(push|fetch)]
    set up remote as a mirror to push to or fetch from

Dell@Admin MINGW64 /g/ManishaGit/test (master)
$ git remote add origin https://github.com/Manisha-atos/gittest.git

Dell@Admin MINGW64 /g/ManishaGit/test (master)
$ git pull origin mater
fatal: couldn't find remote ref mater

Dell@Admin MINGW64 /g/ManishaGit/test (master)
$ git pull origin master
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Unpacking objects: 100% (3/3), 613 bytes | 1024 bytes/s, done.
From https://github.com/Manisha-atos/gittest
* branch master -> FETCH_HEAD
```



- Local → Remote
- \$git push origin master

DevOps Tutorial for Beginners | L x

Manisha-atos/gittest x

Rediffmail x

+

github.com/Manisha-atos/gittest

<> Code

! Issues 0

🔗 Pull requests 0

▶ Actions

📁 Projects 0

📖 Wiki

🛡 Security 0

📊 Insights

⚙ Settings

No description, website, or topics provided.

Edit

Manage topics

🔗 3 commits

🌿 1 branch

📦 0 packages

📢 0 releases

👤 1 contributor

Branch: master ▾

New pull request

Create new file

Upload files

Find file

Clone or download ▾

👤 Manisha-atos Merge branch 'master' of https://github.com/Manisha-atos/gittest

Latest commit d0b6138 5 minutes ago

📄 File1.txt

First Commit

28 minutes ago

📄 readme File1.txt

Create readme.txt

22 minutes ago

📄 readme.txt

🖋

hello
welcome to git

© 2020 GitHub, Inc.

Terms

Privacy

Security

Status

Help

🐙

Contact GitHub

Pricing

API

Training

Blog

About

https://github.com/Manisha-atos/gittest/blob/master/File1.txt

Windows

Edge

Calendar

Chrome

SQL Server

File Explorer

File Explorer

Word

PowerPoint

Excel

Paint

System Tray

11:37 PM

18/05/2020

- Apache Maven Command (software project mgt and compilation tool base on POM –project object model)
- Dependency mgt system
- `mvn archetype:generate`
- `-DgroupId = com.example`
- `-DartifactId = calc`
- `-DarchetypeArtifactId = maven-archetype-quickstart`
- `-DinteractiveMode = false`

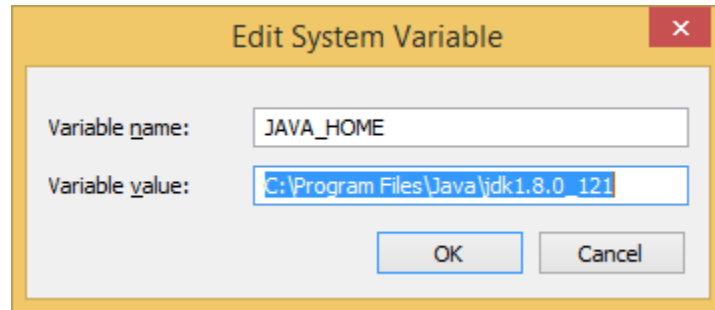
- **MVN package:** This command is used to execute all Maven phases until the package phase. It does the job of compiling, verifying and building the project. It builds the jar file and places it in the specified folder under the specified project.

- **mvn compile:** This command is used to compile the source code. It also compiles the classes that are stored at a particular target or class.
- **mvn clean install:** This maven command helps in executing a clean build life cycle and installs build phase in the default build cycle. This build life cycles may have its build phases and inside each build, there are different build goals. Also, this ensures that the build target is being removed for a new build and adds the clean target.

- **mvn test:** Maven also provides the facility of unit testing particular codes. It runs the tests using suitable [testing frameworks](#).

- **A Build Lifecycle is Made Up of Phases**
- Each of these build lifecycles is defined by a different list of build phases, wherein a build phase represents a stage in the lifecycle.
- For example, the default lifecycle comprises of the following phases (for a complete list of the lifecycle phases, refer to the [Lifecycle Reference](#)):
- validate - validate the project is correct and all necessary information is available
- compile - compile the source code of the project
- test - test the compiled source code using a suitable unit testing framework. These tests should not require the code be packaged or deployed
- package - take the compiled code and package it in its distributable format, such as a JAR.
- verify - run any checks on results of integration tests to ensure quality criteria are met
- install - install the package into the local repository, for use as a dependency in other projects locally

- Set JAVA_HOME to C:\Program Files\Java\jdk1.7.0_60



Environment Variables

Edit System Variable

Variable name: Path

Variable value: I.8.0_121;C:\Program Files\apache-maven-

OK Cancel

Edit System Variable

Variable name: M2_HOME

Variable value: C:\Program Files\apache-maven-3.6.3

OK Cancel

Edit System Variable

Variable name: M2

Variable value: %M2_HOME%\bin

OK Cancel

Edit System Variable

Variable name: MAVEN_HOME

Variable value: C:\Program Files\apache-maven-3.6.3

OK Cancel

Edit System Variable

Variable name: JAVA_HOME

Variable value: C:\Program Files\Java\jdk1.8.0_121

OK Cancel

- Mvn -v
- Create
- Mvn pac
- Mvn clean install
- Mvn validate
- Mvn compile
- Mvn build
- Mvn test

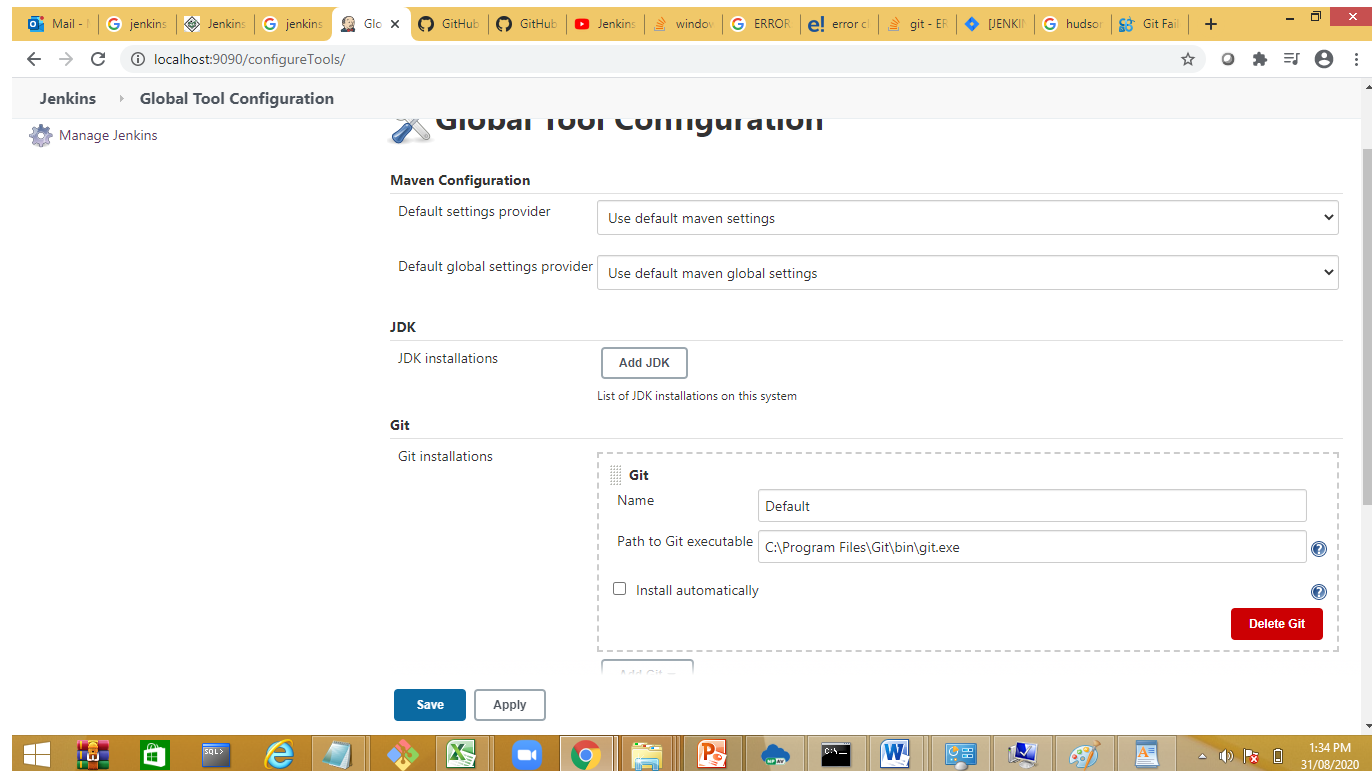
- Jenkins
- 2 ways to run Jenkins
- Standalone
- With web server such as tomcat
-
- <https://updates.jenkins-ci.org/download/war/>
- standalone installation
 -
 - 1.download Jenkins file
 - for standalone we need jdk1.8
 - for tomcat any version
 - 2.copy pate it in drive
 - 3. start a cmd and run the command
 - Java -jar Jenkins.war

- Introduction to Jenkins
- Makes dev.testing deployment easier and faster.
- Bamboo
- Apache Gump
- Jenkins
- Buildbot
- Travis CI

- Check for the github plugin in Jenkins
- Git integration
- Github integration
- Maven integration
- Githubapi

- 1. Install JDK
- 2. Set enviromental for JDK
- 3. Download the Jenkins and in stall
- 4. Run jenkins with local host

- Demo to pull the data from github repository
- Settings



General

Source Code Management

Build Triggers

Build Environment

Build

Post-build Actions

Source Code Management

☐ None

☒ Git

Repositories

Repository URL

Credentials

- none -

 Add

Branches to build

Branch Specifier (blank for 'any')

Repository browser

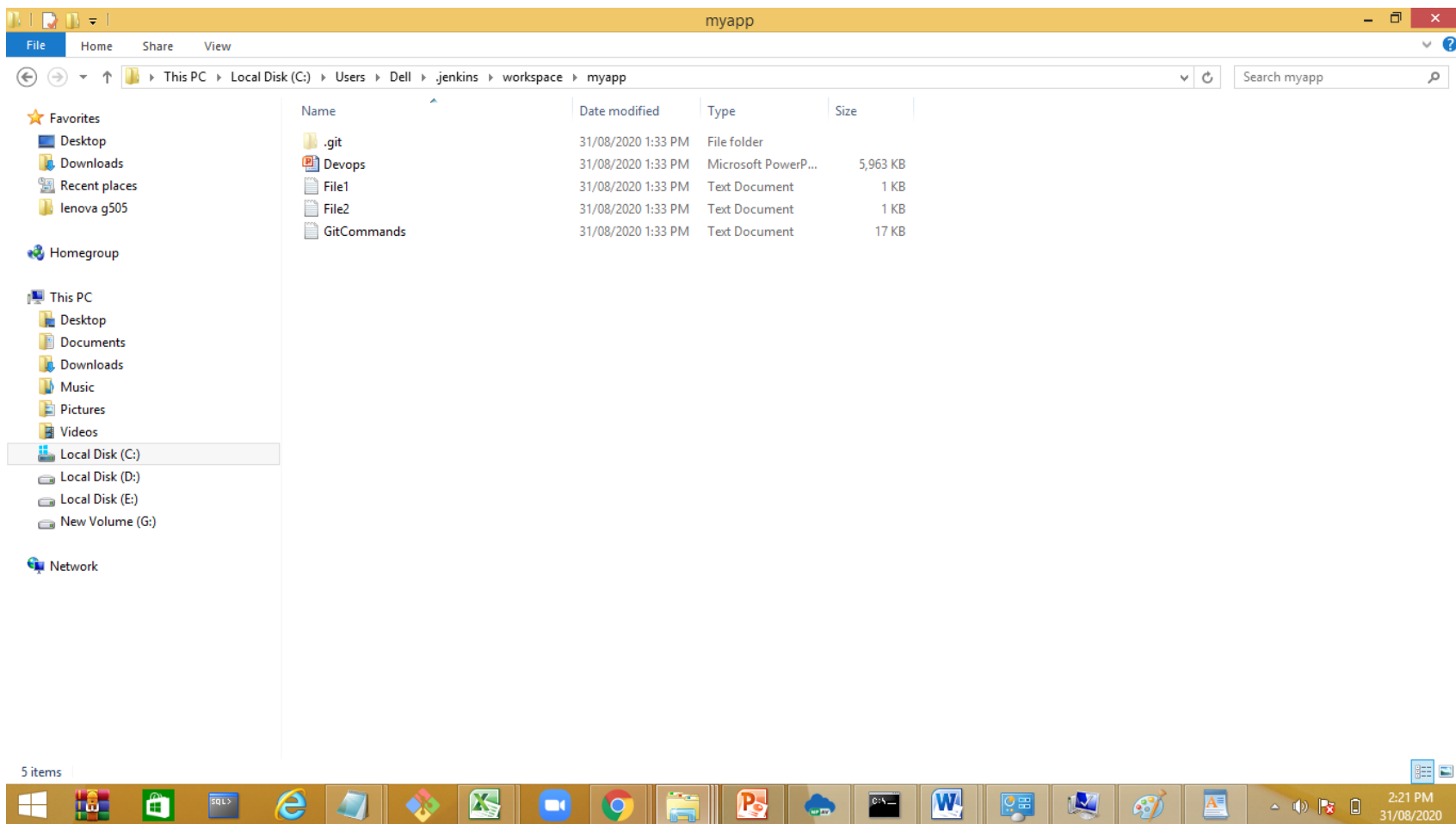
(Auto)

Additional Behaviours

Add

Save

Apply



- Create a new `gitcmd` job to execute the `git` command

Global properties☐ Disable deferred wipeout on this node☒ Environment variables

List of variables

Name PATH

Value C:\Program Files\Git\cmd;C:\Program Files\Git\bin

Add

☒ Tool Locations

List of tool locations

Name (Git) Default

Home C:\Program Files\Git\bin\git.exe

Add

Pipeline Speed/Durability Settings

Pipeline Default Speed/Durability Level

None: use pipeline default (MAX_SURVIVABILITY)

Save

Apply

Mic

Atos

Inte

Con

Get

con

Con

Con

Con

How

how

Build

Jen

how

win

Ma

'run

win

win

localhost:9090/job/gitcmd/configure

☆

⊞

⌵

⌵

⌵

Jenkins

gitcmd

General

Source Code Management

Build Triggers

Build Environment

Build

Post-build Actions

Execute Windows batch command

Command

start "" "%ProgramFiles%\Git\git-bash.exe"
git add .
git commit -m "Commit"
git remote remove origin
git remote add origin https://github.com/Manisha-atos/gittest.git
git push --force origin master

[See the list of available environment variables](#)

Advanced...

Add build step ▾

Post-build Actions

Add post-build action ▾

Save

Apply

Windows

Internet Explorer

Microsoft Edge

Visual Studio Code

File Explorer

Google Chrome

Task View

File Explorer

Microsoft Excel

Microsoft PowerPoint

Microsoft Word

Terminal

File Explorer

Google Chrome

File Explorer

10:52 PM

24/05/2020